UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

DIVISIÓN DE INGENIERÍA ELÉCTRICA

INGENIERÍA EN COMPUTACIÓN

COMPUTACIÓN GRÁFICA e INTERACCIÓN HUMANO COMPUTADORA

MANUAL TÉCNICO

GRUPO: 05

SEMESTRE 2025-2

Objetivo

Diseñar, desarrollar e implementar dos atracciones para formar parte de una feria interactiva con una ambientación y dinámica inspiradas en el videojuego *Ultrakill*, siendo estas un salón de boliche y una jaula de bateo. Estas atracciones mostrarán la ambientación y elementos del nivel “Ship of fools” y del nivel “Cerberus” respectivamente.

Alcance del Proyecto

- Diseño conceptual de ambas atracciones basado en la estética y mecánicas de *Ultrakill*.
- Implementación de la pista de boliche y la jaula de bateo, incluyendo elementos decorativos temáticos.
- Desarrollo de animaciones que se ejecuten al interactuar con las atracciones

Diccionario de funciones y variables:

Boliche

- void renderBuilding: Aplica las transformaciones y renderiza los mesh para construir el edificio del boliche.
- void renderPines: Aplica las transformaciones y renderiza los modelos para los pines del boliche.
- void renderPinesAnimation: Aplica las transformaciones y renderiza los modelos para los pines del boliche, además contiene transformaciones con variables para realizar animaciones.
- void renderTerminal: Aplica las transformaciones y renderiza los modelos para la terminal del boliche.
- void renderMauriceBowling: Aplica las transformaciones y renderiza los modelos para la bola de boliche.
- void renderTable: Aplica las transformaciones y renderiza los modelos para la mesa del boliche.
- void renderChandelier: Aplica las transformaciones y renderiza los modelos para los candelabros del boliche.
- void renderLaneFloor: Aplica las transformaciones y renderiza la mesh para el piso de las líneas de boliche.
- void renderRailing: Aplica las transformaciones y renderiza los modelos para los rieles de las líneas de boliche.
- void renderCarpet: Aplica las transformaciones y renderiza los modelos para la alfombra del boliche.
- void renderChair: Aplica las transformaciones y renderiza los modelos para las sillas de cada mesa dentro del boliche.
- void renderCerberusStatue: Aplica las transformaciones y renderiza los modelos para los estatuas decorativas del boliche.
- void renderBoliche: Función que renderiza los elementos del boliche
    - glm::mat4 model. Variable matriz en donde se almacenan las transformaciones
    - GLuint uniformModel. Variable entera para el uniform model
    - std::vector<Model*> listaModelos. Vector que contiene las direcciones de los objetos modelos que usa el boliche.
    - std::vector<Mesh*> bowlingMeshList. Vector que almacena las mesh creadas para el boliche, es decir, el piso, las líneas, las paredes y el techo.
    - std::vector<Texture*> listaTexturas. Vector con las direcciones de las texturas que utilizan los mesh
    - GLboolean ePressed. Boleano que indica si la tecla E es presionada
    - glm::vec3 modelPosition. Variable que contiene la posición del modelo del avatar
    - GLfloat* bowlingAnimationLane: Es un apuntador hacia un arreglo de números de tipo flotante. Cada de las posiciones representa las variabilidad de las transfromaciones de los elementos de las diferentes líneas de boliche
    - GLfloat deltaTime: Variable de la diferencia entre el frame actual y el anterior utilizada para homologar la velocidad de las animaciones en todos los framerates.

    - GLfloat altura: La altura en la que son posicionados los modelos del boliche
    - GLfloat z: La variable almacena la posición z de los elementos, esta es reducida dentro de un ciclo for para crear varias instancias de una línea de boliche.

Jaula de bateo:

- void renderBattingBuilding: Función que aplica transformaciones renderiza las mesh del edificio donde se encuentran las jaulas de bateo.
- void renderFeedbacker: Función que aplica transformaciones y renderiza el bate
- void renderOrbBall: Función que aplica transformaciones y renderiza la bola.
- void renderLava: Función que aplica transformaciones y renderiza la mesh del piso de lava
- void renderCerberusPitcher: Función que aplica transformaciones y renderiza al pitcher
- void renderReja: Función que aplica transformaciones y renderiza las rejas de la jaula
- void renderV1: Función que aplica transformaciones y renderiza a V1
- void renderBatting: Función que agrupa las funciones de renderizado de los elementos de la jaula de bateo, además de aplicar las animaciones.
    - glm::mat4 model. Variable matriz en donde se almacenan las transformaciones
    - GLuint uniformModel. Variable entera para el uniform model
    - std::vector<Model*> listaModelos. Vector que contiene las direcciones de los objetos modelos que usa el boliche.
    - std::vector<Mesh*> MeshList: Vector que contiene las direcciones de las mesh creadas para el piso, paredes, techo y lava.
    - std::vector<Texture*> listaTexturas. Vector que contiene las direcciones de las texturas utilizadas por las mesh.
    - GLfloat *battingAnimationLane. Variable apuntador hacia un arreglo que contiene la variable usada para modificar las magnitudes de las transformaciones correspondientes a las animaciones
    - GLfloat deltaTime: Variable de la diferencia entre el frame actual y el anterior utilizada para homologar la velocidad de las animaciones en todos los framerates.
    - GLboolean ePressed. Boleano que indica si la tecla E es presionada
    - glm::vec3 modelPosition. Variable que contiene la posición del modelo del avatar
    - GLboolean* battingReverse. Booleano utilizado para invertir la dirección de movimiento de la bola al ser bateada.
    - GLfloat* parryAnimation: Es un apuntador hacia un arreglo de números de tipo flotante. Cada de las posiciones representa las variabilidad de las transfromaciones de los elementos de las diferentes jaulas de bateo
    - GLboolean* parryReverse. Booleano utilizado para indicar que el bate se mueve en dirección opuesta después de batear.
    - GLfloat z: La variable almacena la posición z de los elementos, esta es reducida dentro de un ciclo for para crear varias instancias de una línea de boliche.

Costos

A continuación, se muestra el desglosamiento de actividades con las horas invertidas

FECHA	ACTIVIDADES REALIZADAS	HORAS INVERTIDAS
15/04/2025	Cree el repositorio, subí los archivos necesarios para el proyecto, eliminé archivos que no se ocupan	1
16/04/2025	Agregué modelos de Utrakill, agregué variabilidad a la intensidad de la luz simulando el ciclo de día y noche, agregué textura al suelo y el camino principal. Agregué el avatar con posicionamiento relativo a la cámara, simulando la tercera persona	6
17/04/2025	Agregué el cambio de skybox para el ciclo de día y noche. Agregué más modelos, mesh y texturas para el boliche. Borré archivos no necesarion en el repositorio y realicé el merge de los avances del equipo. Realicé una función para teletransportar la cámara	10
18/04/2025	Moví lo elementos del boliche debajo del mapa, implementé como método de acceso un teletransporte al ingresar al barco.	3

19/04/2025	Se agregaron los rieles y el piso a los carriles del boliche	3
20/04/2025	Agregué los últimos modelos para el boliche, además del puesto de esquites. Implementé variación al ángulo de la luz del sol	5
24/04/2025	Separé al avatar en distintas partes, las acomodé de forma jerárquica e implemente una animación al caminar	4
25/04/2025	Implementé mejoras a la animación de caminar	2
28/04/2025	Agregué la estructura de la jaula de bateo, agregué a V1 y a V2 como NPCs	3
30/04/2025	Agregué las cámaras, agregué interacción con las terminales del boliche para activar la animación	6
07/05/2025	Agregué interacción con las terminales de bateo para activar la animación, arreglé la componente x del sol, agregué el arco con el letrero de texto cambiando, realicé el merge de los avances del equipo	4

En total, se trabajaron 47 horas en el desarrollo del proyecto.

La mano de obra se cobra a MXN$200.00, resultando en MXN$9400.00.

Se agregan los costos de depreciación del equipo de cómputo siendo los siguientes:

	Costo	Vida util (Horas)	Depreciación por hora	Depreciación total
PC	18,000	60480	$0.30	$13.99
Monitor	4000	60480	$0.07	$3.11
Monitor	10000	60480	$0.17	$7.77
Teclado	800	30240	$0.03	$1.24
Mouse	800	30240	$0.03	$1.24
Internet	900	840	$1.07	$50.36

Resultando en un total de MXN$77.71

Se suma también el costo de la energía eléctrica. Considerando un consumo de 0.2 kWh y un costo de MXN$2.00 por kWh, dando un total de MXN$18.8

Se obtiene un subtotal de MXN$9,496.51, a lo cual se agrega un margen de ganancia de 30%, resultando en un costo de MXN $12,345.47

Finalmente, se agrega el 16% de IVA, obteniendo el total de MXN $14,320.74

Conclusiones

Al finalizar el desarrollo del proyecto, se cumplió el objetivo de crear dos atracciones interactivas de feria con temática de *Ultrakill*, integradas en un entorno digital tridimensional. Estas atracciones permiten al usuario explorar el espacio e interactuar con mecánicas inspiradas en el juego, como el parry y la activación de elementos dentro del escenario. El resultado ofrece una experiencia interactiva que combina referencias visuales del juego con una ambientación diseñada para incentivar la exploración.